X-Ray Spectrometer Instrumentation with a Personal Computer

Skaarup, P.; Vogeley, E.

Publication date:
1989

Document Version
Publisher's PDF, also known as Version of record

Risø-M-2733

RISØ

Risø-M-2733

X-Ray Spectrometer Instrumentation with a Personal Computer

Per Skaarup and Erik Vogely

Risø National Laboratory, DK-4000 Roskilde, Denmark
January 1989

RISØ-M-2733

X-RAY SPECTROMETER INSTRUMENTATION WITH A PERSONAL COMPUTER

Per Skaarup and Erik Vogeley

Abstract. A description is given of an instrumentation for control of an X-ray spectrometer used in solid state physics experiments. The instrumentation includes a personal computer (PC) and a European Computer Bus (ECB) interface system. Details are given of the operating software.

January 1989
Risø National Laboratory, DK 4000 Roskilde, Denmark

ISBN 87-550-1448-8
ISSN 0418-6435

Grafisk Service Risø 1989

TABLE OF CONTENTS

1. INTRODUCTION

During recent years groups at Risø have used X-ray scattering in their research work. Three Risø X-ray spectrometers are available at present, one located at a rotating anode X-ray generator at Risø, the two others located at the Desy synchroton in Hamburg. This report describes a new instrumentation based on a personal computer for one of these spectrometers. The instrumentation has been in operation since February 1988.

2. GENERAL DESCRIPTION

The spectrometer may mechanically be set up in several ways. In one type of experiments, as shown in Fig. 1, crystals are mounted on the monochrometer table, the sample table, and the analyser table. X-rays diffracted by the crystals are counted in a detector on the detector arm. The crystals are mounted on goniometers so that their orientation can be adjusted. The movement of the goniometers as well as the spectrometer axes is controlled by stepping motors. The gearing is such that a precision of 0.001° is achieved.

The high rate of data acquisition and the complexity of the control of the many axes call for a computer controlled instrumentation. Former spectrometer instrumentations at Risø have been based on PDP-11 computers and CAMAC interface systems (1). But to utilise the low-cost alternative offered by personal computers (PCs) and cheaper interface systems, this new instrumentation is based on an IBM/PC/XT type of personal computer and a Z80 based interface system called the ECB system. The connection between the PC and the ECB system is through a GPIB (IEEE-488) standard bus. The control of the spectrometer axes is accomplished through ECB motor control modules connected

to stepping motor drives. The ECB rack also contains scalers, timer, display drive module, modules for read and control of a digital voltmeter with scanner, and the Z80 control module with RAM/ROM storage. The nuclear channels with amplifiers, single channel analysers, and supply for detectors are contained in a NIM bin. A block diagram of the instrumentation is shown in Fig. 2. The photo of the instrumentation in Fig. 3 shows that the PC, the NIM bin, and the various boxes are placed at the experimenter's table. These are all units with keyboard and buttons that the experimenter needs in order to control the spectrometer. The other units, ECB rack and motordrive racks, do not normally require the experimenter's attention and are located in a cabinet.

The computer runs under the DOS operating system with the user software written in FORTRAN-77. The user need not know FORTRAN-77, however. His interaction with the system is through a general spectrometer program called TASCOM which in a user friendly way allows input of parameters for the experiment, specification of scans, and storage and presentation of the measured data.

A typical measuring procedure for the spectrometer has the following sequence of operations: calculation of the motor positions, adjustment of the spectrometer axes to the new positions, resetting of scalers and start of counting for a preset time or count, reading-out of measured data and axes position, presentation on screen and printer of the data, and storing of the data on disk.

Data reduction and evaluation of measured data may be performed on the system itself. Data may be transported to other personal computers on DOS formatted floppy discs, or to other types of computers with the KERMIT communication program that transmits the data through the PC serial port connected to the Risø datanet.

3. THE ECB SYSTEM AND THE PC

The ECB system is a de facto industrial standard based on the ZILOG Z80 microprocessor and its bus. The ECB system is mechanically housed in an EURO rack with the Z80 bus on the backplane. The functional modules are in single eurocard format and plugs into 64 pole connectors on the backplane. Z80 is an 8 bit processor and the bus allows 8 bit data transfer between the ECB modules. In addition, there are 16 adress lines and a number of control and power lines on the ECB backplane.

When connected to a PC through a GPIB interface the ECB system is configurated with the Z80 module, the GPIB module, and the functional modules. The Z80 module contains Z80 processor, RAM and ROM memory, and a serial port. In the ROM memory the program that controls all activity in the ECB system is loaded. Basically, this program is a GPIB driver for the communication with the PC through the GPIB interface, and a number of application specific subroutines that may be called from the PC. In addition, for the X-ray spectrometer there are display update routines that are activated by an internal ECB timer. The program is written in Z80 assembly code but it is planned to rewrite it in C language.

The PC is an IBM/XT compatible Olivetti M240 with a Winchester disc and floppy disc. Communication with the ECB system is through an IBM/GPIB board plugged in the PC using the GPIB driver supplied with the board.

Also plugged in the PC is a multichannel analyser board from Nucleus Inc. which enables the PC to collect a pulse height distribution spectrum from a detector. Such spectra may be collected with the PC in a menu-driven mode or as a part of the TASCOM general spectrometer program.

4. DETAILED DESCRIPTION OF THE INSTRUMENTATION

4.1. Motor Control System

All motors used are unipolar SLO-SYN stepping motors type M061-FD08 with a maximum speed of 3000 steps/sec or type M092-FD09 with a maximum speed of 1200 steps/sec. The motors are controlled from ECB-module P1648*), connected to motor drives P1604. Manual motor control is achieved with ECB module P1814 and manual control box P1811.

Stepping motor control ECB module P1648. Each P1648 module can control four stepping motors either by order from the ECB bus or by signals on a manual control connector on the module. There are six P1648 modules in the instrumentation, thus 24 motors may be controlled.

A P1648 module contains four microprocessors, Intel 8742, one for each motor controlled by the module. The Intel 8742 has 2K ROM memory that holds the program for the motor control, and 128 RAM registers for motor parameters, number of steps, etc. The RAM registers may be loaded or read via the ECB bus.

One 24 bit RAM register keeps track of the motor position as it acts as an up/down counter of the steps supplied from the Intel 8742 to the motor drive.

Another 24 bit RAM register controls the stepping of the motor. This register is loaded from the ECB bus with the number of steps necessary to move the motor to the position wanted. The register is then counted down towards zero while stepping pulses are generated to the motor drive. The frequency of the pulses decides the stepping rate of the motor.

---

*) P-numbers refer to Risø constructions.

At a high stepping rate the motor must be accelerated to and decelerated from the stepping rate in order to ensure that no steps are lost during start or stop of the motor. The acceleration and deceleration are included in the motor control program in the Intel 8742 ROM memory, and parameters such as time constant for acceleration and deceleration, start/stop speed, maximum speed, etc. may be loaded from the ECB bus to the RAM registers.

The motor control program reacts on limit switch signals for clockwise or counterclockwise rotation and on an error signal. The two first signals will disable the stepping signals for one direction of rotation but allow the other direction, while the error signal disables stepping in both directions. At a high stepping rate the motor is decelerated before the stepping signals are disabled.

The motor control program generates control signals in connection with a motor run. The signals are used to enable/disable motor drive power and to control air bearings on the spectrometer axes.

A more detailed description of P1648 is found in (2).

Manual motor control P1811 and P1814. The motor control program in the ROM memory of Intel 8742 on P1648 has a manual mode which may be activated with signals on a connector for that purpose on P1648. These signals derive from manual motor box P1811 connected to ECB module P1814 which is connected to the six P1648s in the rack. With a 24 position switch on the box one motor is selected for manual operation and the motor is run with a joystick on the box.

The joystick generates an analog voltage which on module P1814 is converted to a digital value for the speed and direction of the motor. Furthermore, on the box two speed ranges, low or high, may be selected. Thus the motor may be run either at low speeds, e.g. between 1 and 20 steps/sec. or at high speed, e.g., between 200 and 3000 steps/sec. This scheme allows the user to drive a motor with high speed close to a wanted position and then with low speed to the exact position.

When a motor is under manual control, it is not possible to control it with orders from the ECB bus. The position RAM register for the motor is, however, updated when the motor is moving and may be read via the ECB bus. In manual mode the limit switch signals and the error signals disable stepping as they do when the motor is running by ECB orders.

Stepping motor drive. The stepping pulses from P1648 are fed to P1604 motor drives. Each motor has its own drive, but eight drives in a rack share the power unit. Thus only one of the eight motors connected to the rack may run at a time. The power unit is allocated to a particular drive with the enable signal which is asserted from P1648 before the stepping pulses are sent. As a result of one drive being enabled an inhibit signal common for all the drives in a rack is set and transmitted back through all the drives in the rack to the controlling P1648 modules. This prevents more than one motor control from asking for the power unit in a drive rack.

When manual control is selected for a motor, the matching drive will be enabled. In this case it is not possible to run the motors connected to the seven other drives in the rack.

The step pattern applied from the drive to the motor is changed for each stepping pulse received by the drive. When the drive is disabled, the pattern is saved, and when the drive is reenabled, the old pattern is applied again. Thus no motors loose steps when the power unit is multiplexed among the drives in the rack.

The power unit can deliver 6 amps per phase through 4.5 ohm dropping resistors contained in the rack.

The signals from limit switches are connected to the P1815 Limit Switch module where they are integrated to suppress noise. At present, input of individual limit switch signals for each motor is not used on this spectrometer. Instead all limit switches are connected in a chain through a P1464 limit switch box and the resulting signal is fed to the error input of P1815. The error is

passed to the limit switch modules in the other motor drive racks and via the back planes and the motor drive modules to all ECB motor control modules. Thus any limit switch action will stop all running motors.

The error signal may be bypassed with a push button on the limit switch modules. This allows the user to run a motor away from an activated limit switch while the error signal is temporarily bypassed. Another push button on the limit switch modules sets the error signal. This may be used as a manual emergency stop of the motors.

The scheme used here with all limit switches chain connected to the error input has been found simpler to implement at this spectrometer compared with a scheme where two individual limit switch signals for each motor are connected to the ECB system. A combination of the two schemes is possible where some of the motors have individual limit switch signals sent to the ECB system while others use the chain method.

A more detailed description off the motor drive system is found in (3).

For future instrumentations a new motor drive system is at present being developed. The new drives are more powerful than the present ones and are based on a microstep scheme giving a softer motor run without resonance problems at certain speeds. They will be located at the spectrometer in groups close to the motors they run. The new drive will use the same input signals as the present ones so that both types may be operated from P1648.

4.2. Display System

The display system consists of an ECB video driver module and one or two 12" video monitors. If two monitors are used, they are connected in series showing the same picture. The display is used as an alphanumeric display showing positions of motors, contents of scalers and timer, and count rate in scalers. The characters displayed are of a large size, 2 cm x 2.5 cm on a 12" monitor, so

that the display may be read from a distance of up to 10 metres. With such large characters four lines only can be displayed at a time. On each line one selected unit (motor, scaler, etc.) is shown.

The display is updated 12 times per second. Each update is initiated by interrupt from a timer used for that purpose on the P1834 ECB module. The update is started at power on and continues independently of the PC so that the display is always active.

The selection of the units to display on each of the four lines may be done by order from the PC or by use of the display scroll box P1865. The scroll box is connected to the serial port of the ECB Z80-CPU module and works independently of the PC. With push buttons on the box for forward and backward scroll the user can select the unit to display on a chosen display line from the unit table in the Z80 program.

If manual motor control is selected for a motor by means of the manual motor box, the position of that motor is shown on line # 1 at the display overwriting the unit selected by the scroll box. To indicate manual motor control the position is shown underlined.

An example of a display picture is shown in Fig. 4.

4.3. Counting System

The ECB scaler/timer modules comprise one 16 bit timer and three 24 bit scalers. An 8 bit prescaler may be attached to each scaler. The timer or one of the scalers may be specified as a preset unit in a measurement.

The timer and scalers are controlled with orders from the ECB bus. Read-out of scalers and timer is performed in bytes of 8 bits. Therefore, reading the scalers and timer while they are counting may produce wrong results.

4.4. Temperature Control System

Automatic setting of the temperature control is done by means of a stepping motor driven reference potentiometer in a proportional/integral regulation. The motor is run from one of the stepping motor drives.

A Solatron digital voltmeter type 7075 with scanner is used to read up to ten temperature sensors and is interfaced through two parallel input/output ECB modules.

5. SOFTWARE

The software for this instrumentation comprises the ECB system software, the PC/GPIB driver, the subroutines for control of motors and scalers, and the TASCOM program.

5.1. ECB System Software

The ECB system software is written in Z80 assembler and held in ROM memory on the ECB CPU module. It contains the GPIB communication routines (4) and a number of application specific subroutines that can be activated by calls through the GPIB.

There are three main types of GPIB communication calls. In the first type a specified number of bytes are sent from the GPIB to specified addresses in the ECB bus. In the second type a specified number of bytes are read from a specified ECB address to the GPIB. The third and most often used call is the function call which calls one of the application specific subroutines. Before the subroutine call four bytes are sent to the D, E, B, and C registers of the Z80, and after the call D, E, B, and C registers are returned. Each function call is identified by a

function number in the range of F=128-196. All the function calls are listed in Appendix A.

An example of a function call is F=138, read a timer or scaler. Prior to the subroutine call the Z80 registers D, E, B, and C are loaded from the GPIB bus. Register C is used by the subroutine to specify timer (C=0) or one of the scalers (C=1-3). The subroutine reads the scaler/timer ECB module and places the three bytes read in registers D, E, and B. Following the subroutine call the contents of registers, D, E, B, and C is sent to the GPIB port.

Also the display update program is part of the Z80 software. The program is activated by interrupts from P1834. It consumes approximately 25% of the Z80 CPU time. If very fast response to GPIB function call is wanted, the display program may be disabled during such operations and later reenabled, but normally the display program is always running.

When power is removed from the ECB rack motor positions as held in P1648 modules are stored in non-volatile RAM together with other parameters. At power on, these values are reinstalled and the display program started. It is thus possible to run the motors with the manual motor control box and observe the motor positions as well as the count rate in the scalers on the display even if the PC is not connected to the ECB system.

5.2. PC/GPIB Driver

This is the GPIB communication software in the PC. It works under the DOS operating system and is written in FORTRAN-77 using the basic IBM/GPIB assembler written driver routines. The calls to the ECB system must match in number and type of bytes sent or received over the GPIB with the corresponding Z80 communication routines described above.

The calls are in a general form, e.g., the routine that sends a number of bytes to a given address in the ECB system is called

```
CALL GPIBWRIT (DRIVER, UNIT, ADDRESS, NUMBYTE)
```

Here DRIVER is a number between 0 and 7 depending on the switch setting on the GPIB board in the PC. In the spectrometer-PC only one board is used, hence DRIVER=0. Several devices, e.g., two ECB systems, might be on the same GPIB bus. The parameter UNIT is used to distinguish between devices on the GPIB bus. The bytes to be sent are held in a common character array.

The function call is

```
CALL GPIBFUNC (DRIVER, UNIT, FUNCTION)
```

where FUNCTION is the function number as listed in Appendix A. The four bytes sent with the function call and the four bytes returned from the ECB system are stored in a common character array DATAF.

To read the contents of a scaler first set DATAF(4) = scaler number. Then do the function call with FUNCTION = 138. After the call the three bytes read are held in DATAF(1), DATAF(2), and DATAF(3).

Included in the driver are two calls that are not directed specifically to the ECB system but to other types of GPIB deviced, e.g., a digital voltmeter with GPIB interface. These calls are

```
CALL GREAD (DRIVER, UNIT, NUMBYTE)
CALL GWRIT (DRIVER, UNIT, NUMBYTE)
```

The calls read or write the specified number of bytes to/from the device given with the UNIT parameter. The bytes are received by/sent from a common character array GDATA.

5.3. Application Subroutines

This is a set of FORTRAN-77 subroutines used to control and read scalers, motors, digital voltmeter, etc. They communicate with

the ECB system by means of calls to the PC/GPIP driver. A list of the subroutines is found in Appendix B.

To read the contents of a scaler call the read-scaler-timer subroutine

```
CALL READST (STUNIT, VALUE)
```

with the parameter STUNIT = scaler number. The routine returns the scaler contents in VALUE. Internally, the routine does a GPIB function call as described in 5.2. and a subsequent calculation of VALUE.

5.4. TASCOM

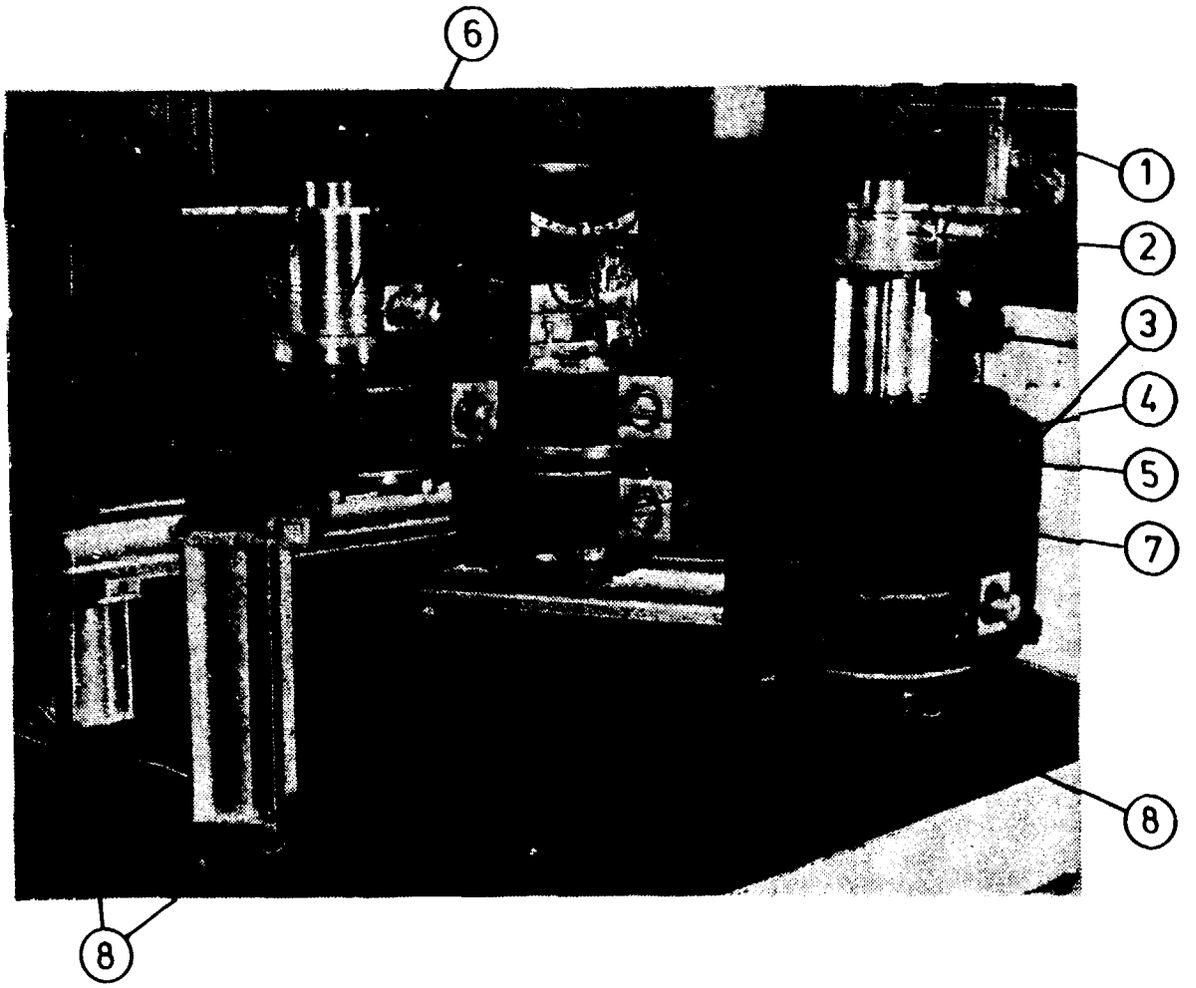
The TASCOM program is the normal user interface to the spectrometer. It has a number of BASIC-like commands that allow easy specification of the operations wanted. For instance,

```
M1 = 10000, M2=20000 <return>
```

moves motor #1 to position 10000 and motor #2 to position 20000. Other commands operate and read scalers, specify output format on printer and on files, or create program loops and branches (IF-THEN, etc.).

The commands may be given in interactive mode from the PC keyboard, or they may be combined to form small programs saved on a disc file. A number of such prewritten programs are available for various scan types with the spectrometer.

TASCOM is written by Collin Broholm, Physics Department, Risø National Laboratory.



1. Beam tube from X-ray generator.
2. Monochromator table.
3. Monochromator arm.
4. Sample arm.
5. Sample table.
6. Analyser table.
7. Detector arm.
8. Air bearings.

Fig. 1. X-ray spectrometer.

DIAGRAM OF RØS 4

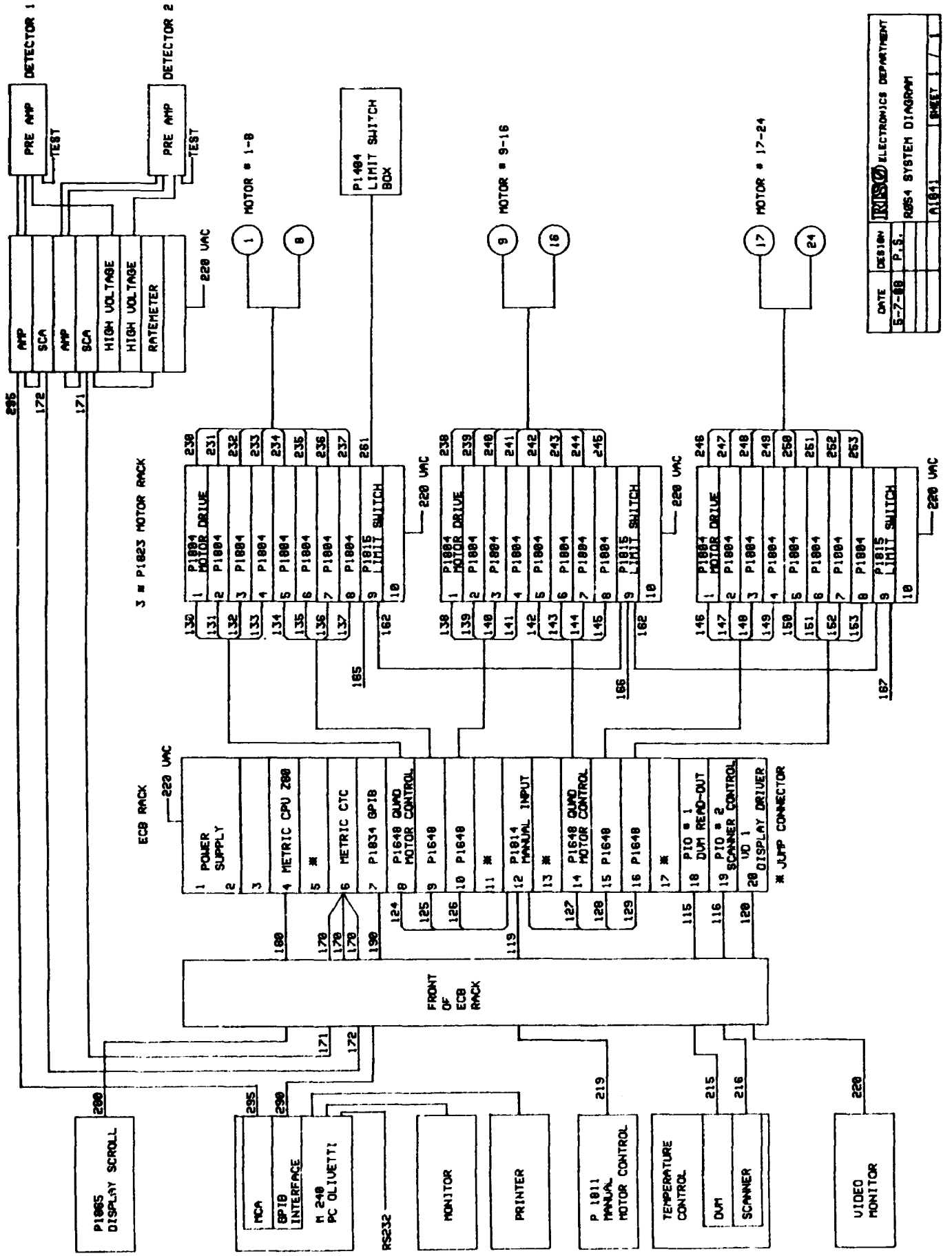


Figure 2.

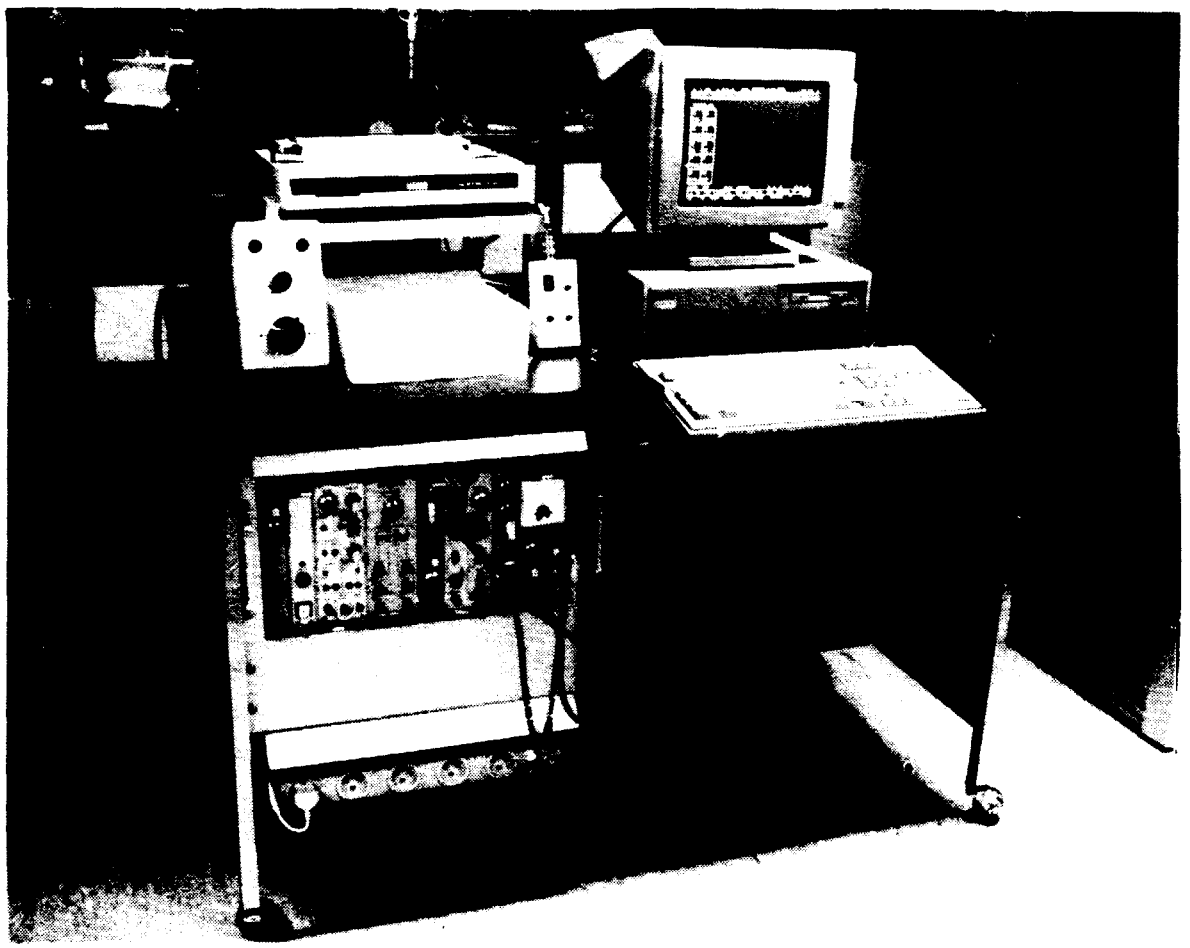


Figure 3. Photo of part of the instrumentation.

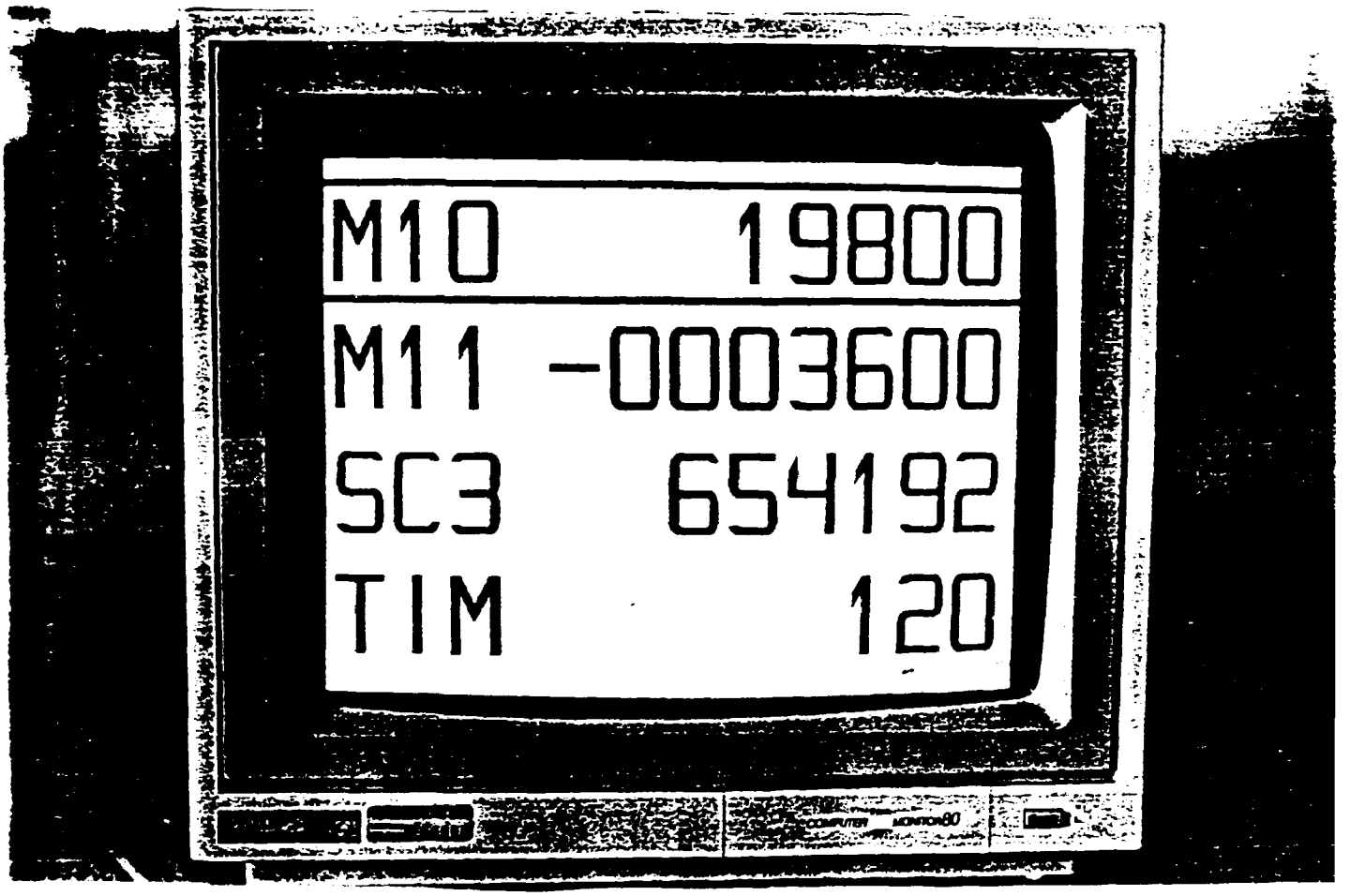


Fig. 4

Photo of the display approx. 1/3 of natural size.

Motor 10 is under manual control as indicated by the underlining and in position 19.800 degree. Motor 11 is in positi -3.600 degree. Contents of scaler 3 and the timer is 654198 counts and 120 sec., respectively.

APPENDIX A.

List of Z80 ECB subroutines callable with function calls from FORTRAN

Where functions use input to Z80 registers D, E, B, C or where functions return D, E, B, C this is noted.

FUNCTION

F=128 ENABLE DISPLAY

F=129 DISABLE DISPLAY

F=130 LOAD DISPLAY TABLE
Input: D,E,B,C: Display unit numbers for lines 1-4.

F=131 CLEAR DISPLAY

F=132 SET TIMER FREQUENCY
Input: D=1 for 1 Hz, D=10 for 10 Hz

F=133 CLEAR AND PRESET TIMER/SCALERS
Input: D,E,B: Preset value
C=0: Preset timer, C=1, 2, or 3:
Preset scaler 1, 2, or 3

F=134 CLEAR TIMER AND SCALERS

F=135 START TIMER AND SCALERS
Input: D=1 Interrupt at preset stop, D=0 No interrupt

F=136 STOP TIMER AND SCALERS

F=137 READ START/STOP FLAG
Return: D=1 for scaler/timer counting
E,B,C: Bits 1-24 on for motors 1-24 running

F=138 READ THE TIMER OR A SCALER
Input: C=0 for timer, C=1, 2, or 3 for scaler 1, 2, or 3
Return: D,E,B: Scaler/timer contents

F=139 LOAD PRESCALERS
Input: D,E,B: Prescal count for scalers 1, 2, or 3,
respectively.

F=140 LOAD MOTOR PARAMETER
Input: C= motor number, B= command code,
E= parameter value
Return: C=0: Error, then B= Error code

F=141 LOAD NUMBER OF MOTOR STEPS AND START THE MOTOR
Input: C= motor number, D,E,B: number of steps to run
Return: C=0: Error, then B= Error code

F=142 LOAD MOTOR POSITION
Input: C= motor number, D,E,B: motor position
Return: C=0: Error

F=143 SAVE MOTOR POSITIONS IN NON-VOLATILE RAM MEMORY

F=144 READ MOTOR STATUS
Input: C= motor number, B= command code
Return: B: Status byte, C=0: Error

F=145 READ MOTOR POSITION
Input: C= motor number
Return: C=0: Error, C=+1/-1: Sign of motor position
D,E,B: Motor position

F=146 LOAD MOTOR CONTROL BYTE
Input: D= motor control byte

F=147 READ DIGITAL VOLTMETER
Input: D= channel number
Return: DE: Address in RAM memory of data read

F=148

F=149 Reserved (SANS)

F=150 Reserved (SANS)

F=151 READ PRESCALER SETTING

F=152 LOAD APPLICATION SPECIFIC PARAMETERS
INPUT: B=APPLICATION, D=NUMBER OF MOTORS, E=CPU-TYPE

F=153 DISABLE REQUESTS

F=154 ENABLE REQUESTS

F=155 RESTORE MOTOR POSITIONS FROM NON-VOLATILE RAM MEMORY

F=156 ENABLE MANUAL MOTOR DISPLAY FEATURE

F=157 DISABLE MANUAL MOTOR DISPLAY FEATURE

F=158

F=159 GET ADDRESS OF MOTOR PARAMETERS IN NON-VOLATILE RAM MEMORY
Return: DE: Address

F=160

F=161 ENABLE MANUAL MOTOR CONTROL OF THE MOTORS

F=162 DISABLE MANUAL MOTOR CONTROL OF THE MOTORS

F=163 LOAD RATEMETER MODE FOR DISPLAY
Input: D,E,B,C: Mode numbers for lines 1-4 on display

APPENDIX B.

List of FORTRAN-77 application subroutines

```
     SUBROUTINE MPOSREAD(MOTOR,POSITION,ERROR)
Read the position of a motor from the ECB module.
Fortran call:
     CALL MPOSREAD(MOTOR,POSITION,ERROR)
ERROR return:  =  0   call was successful
               = -1   wrong motor number
               =  1   position read error
```

```
     SUBROUTINE MPOSWRIT(MOTOR,POSITION,ERROR)
Write the position of a motor to the ECB module
Fortran call:
     CALL MPOSWRITE(MOTOR,POSITION,ERROR)
-2**23 < POSITION < 2**23÷1
ERROR return:  =  0   call was successful
               = -1   wrong motor number
               =  1   write error
```

```
     SUBROUTINE MPARWRIT(MOTOR,COMMAND,PARAMETER,ERROR)
Write a command and the corresponding parameter to motor
module P1648. Legal commands are:
 COMMAND   PARAMETER          MEANING
     3       0 - 1          Disable/enable manual control
     4       0 - 63         Slow step speed
     5       0 - 63         Fast step speed
     6       0 - 63         Speed at start of acc.
     7       1, 2, or 3     Acceleration/deceleration time
     8       1 - 255        Delay
     9                      Motor control, start/stop of motor
    10       0 - 63         Manual slow speed
ERROR return:  =  0   call was successful
               = -1   wrong motor number
               =  1   write error
               =  2   illegal command
               =128   motor under manual control
```

```
      SUBROUTINE MSET(MOTOR,ERROR)
Set a motor to a new position.
      MOTOR = motor number
      ERROR return:  =  0   call was successful
                     = -1   wrong motor number
                     =  1   old position read error
                     =  2   motor already running
                     =  3   start command error
                     =  5   inhibit from motor drive
                     = 10   limit switch + and - on
                     = 11   limit switch - on
                     = 12   limit switch + on
                     =128   motor in manual mode
      COMMON array elements:
          NEWPOS(MOTOR)= wanted new motor position
          SPEED(MOTOR) =0 for low speed, =1 for high speed
          BACKLA(MOTOR)= steps for backlash compensation
          MON(MOTOR)= loaded by MSET with number of steps at start


      SUBROUTINE MTESTRUN(MOTOR,RUN,ERROR)
Test if a motor is running.
      MOTOR= motor number
      RUN: returns 0 for not running, 1 for running
      ERROR return and COMMON array elements:
        The same as for subroutine MSET.


      SUBROUTINE MSTATUS(MOTOR,STATUS)
Read motor status.
      MOTOR= motor number
      STATUS return:  =  0  none of the situations below
                      = -1  wrong motor number
                      =  2  motor running
                      =  5  inhibit from motor drive
                      = 10  limit switch + and - on
                      = 11  limit switch - on
                      = 12  limit switch + on
                      =128  motor under manual control
```

SUBROUTINE ENMMAN

Enable motor manual control box.

SUBROUTINE DISMAN

Disable motor manual control box.

SUBROUTINE CLEANUP

Clean-up before returning to monitor:

Stop all motors.

SUBROUTINE MPARSTOR(MOTOR)

Store motor parameters in the ECB system non-volatile memory.

MOTOR = Motor number

The six motor parameters for MOTOR are held in common array MPAR

SUBROUTINE CLEARS

Clear scalers and timer.

After a call to clear, preset values must be reloaded while prescaler values are unchanged.

SUBROUTINE STARTS(REQUEST)

Start scaler and timer without clearing them first.

REQUEST=1 enables an interrupt request to be generated when scaler/timer stops, REQUEST=0 does not

SUBROUTINE STOPS

Stop scalers and timer.

```
     SUBROUTINE PRESET(STUNIT,PREVAL,ERROR)
```
Clear timer and scalers and the old preset value, then preset a timer or a scaler to PREVAL.
STUNIT = 0: timer, STUNIT = 1, 2, or 3: scalers 1, 2, or 3
0 < PREVAL < 2**24
ERROR Return: = 0: No error
= 1: Preset value less than 1
= 2: Wrong STUNIT number

```
     SUBROUTINE READST(STUNIT,VALUE)
```
Read scaler/timer.
The contents of scalers and timer is not changed.
Read-out while counting is allowed but may result in wrong read-out values.
STUNIT=0: timer, STUNIT=1, 2, or 3: scalers 1, 2, or 3

```
     SUBROUTINE PRESCA(PRESC1,PRESC2,PRESC3)
```
Load prescaler values to scaler 1-3.
0 <= PRESCn < 255

```
     SUBROUTINE TIMFRQ(FREQ)
```
Set the frequency of the timer to either 1 Hz or 10 Hz.

```
     SUBROUTINE SCTEST(COUNT)
```
Test if scalers and timer are counting.
Returns COUNT = TRUE if counting and
COUNT = FALSE if not counting.

```
     SUBROUTINE RATMOD(LINE1,LINE2,LINE3,LINE4)
```
Load line-mode values for ratemeter.
The line-mode values specify the integration time constant for the ratemeters displayed on the four lines of the display.
Line-mode values = 0: no integration
= 1: 1 sec integration
= 2: 4 sec integration

```
      SUBROUTINE DVM(CHANNEL,VALUE)
```

Read the voltage in channel CHANNEL of the DVM.
CHANNEL is the channel number on the scanner. CHANNEL also specifies the integration time for the DVM:

```
      1<=CHANNEL<=9      Channels 1-9 with 1.0 sec integration
  128+1<=CHANNEL<=9+128 Channels 1-9 with 0.1 sec integration
```

The DVM reading is returned in the real parameter VALUE.

```
      SUBROUTINE APPPAR (APPLIC, CPUTYP, NUMMOT)
```

Load application specific parameters

```
      APPLIC = 1 for SANS instrumentation
               2 for X-ray instrumentation
               3 for TAS3 instrumentation
               4 for ultra sonic instrumentation
      CPUTYP = 0 WP-CPU
               1 Metric-CPU
      NUMMOT = Number of motors in the instrumentation
               4< = NUMMOT < = 24
```

REFERENCES

1) Per Skaarup: Instrumentation For an X-Ray Spectrometer. Risø-M-2184, July 1979.

2) Jørgen Bundgaard, Erik Hansen and Per Skaarup: Stepping motor control ECB module. Risø-M-2750, 1989.

3) Jørgen Bundgaard, John Olsen and Per Skaarup: Stepping motor drive system, Risø-M-2734, 1988.

4) Jørgen Bundgaard: The GPIB Drivers, Internal Report, Risø 1984.

5) Colling Broholm: TASCOM user guide, Internal report, 1986.

Risø National Laboratory **Risø – M – 2733**

Title and author(s)

X-RAY SPECTROMETER INSTRUMENTATION
WITH A PERSONAL COMPUTER

Per Skaarup and Erik Vogeley

Date January 1989

Department or group

Electronics

Groups own registration number(s)

R02-88

Project/contract no.

Pages 31 **Tables** **Illustrations** 4 **References** 5

ISBN 87-550-1448-8

Abstract (Max. 2000 char.)

A description is given of an instrumentation for control of an X-ray spectrometer used in solid state physics experiments. The instrumentation includes a personal computer (PC) and a European Computer Bus (ECB) interface system. Details are given of the operating software.

Descriptors

MICROCOMPUTERS; ON-LINE CONTROL SYSTEMS; SOLID STATE PHYSICS;
X-RAY SPECTROMETERS

**Available on request from Risø Library, Risø National Laboratory, (Risø Bibliotek, Forskningscenter Risø),
P.O. Box 49, DK-4000 Roskilde, Denmark.
Telephone 02 37 12 12, ext. 2262. Telex: 43116, Telefax: 02 36 06 09**

Available on exchange from:
Risø Library,
Risø National Laboratory,
P.O. Box 49, DK-4000 Roskilde, Denmark
Phone 42 37 12 12, ext. 2208/2269,
Telex 43 116, Telefax 46 75 56 25

ISBN 87-550-1448-8
ISSN 0418-6435